Algorithmic Motion Planning (236610)
Lecture 3—Arrangements & Pianos

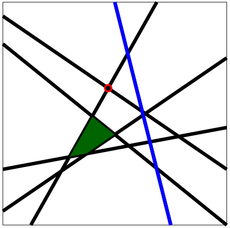
Oren Salzman

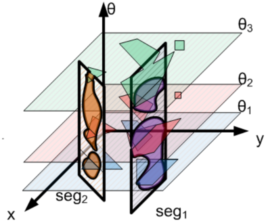
Computer Science Department, Technion

Today's lecture

The piano-movers problem—general exact algorithms for motion planning

- Arrangements
- Arrangements & C-spaces
- General exact algorithms for motion planning





This lecture (and most figs) are based on CG* book, Ch. 28 (Arrangements) and Ch. 50 (Algorithmic Motion Planning) as well as slides by Dan Halperin

*M. de Berg, O. Cheong, M. van Kreveld, M. Overmars: *Computational geometry: algorithms and applications*, 3rd Ed. Springer 08

Arrangements

Definition

Given a finite collection S of geometric objects such as hyperplanes or spheres in $\mathbb{R}^d$, the arrangement $\mathcal{A}(S)$ is the decomposition of $\mathbb{R}^d$ into connected open cells of dimensions $0, 1, \ldots, d$ induced by S.

- Objects can be linear (lines, segments) or non linear (parabolas, rational functions), bounded (circles) or unbounded (hyper-planes)
- Have applications in robotics, CAD / CAM and even the formation and dynamics of the Cosmic Web.

Example—Arrangements of lines

Definition

Given a finite collection S of geometric objects such as hyperplanes or spheres in $\mathbb{R}^d$, the arrangement $\mathcal{A}(S)$ is the decomposition of $\mathbb{R}^d$ into connected open cells of dimensions $0, 1, \ldots, d$ induced by S.

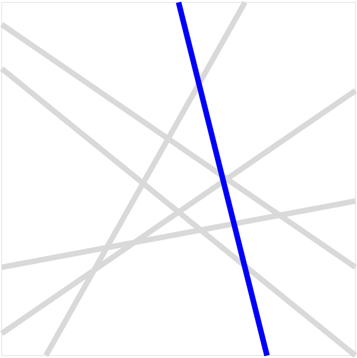
- S is a set of n lines in $\mathbb{R}^2$
- Cells of dimension 0 are points
- Cells of dimension 1 are segments
- Cells of dimension 2 are polygons

Example—Arrangements of lines

Definition

Given a finite collection S of geometric objects such as hyperplanes or spheres in $\mathbb{R}^d$, the arrangement $\mathcal{A}(S)$ is the decomposition of $\mathbb{R}^d$ into connected open cells of dimensions $0, 1, \ldots, d$ induced by S.

- S is a set of n lines in $\mathbb{R}^2$
- Cells of dimension 0 are points
- Cells of dimension 1 are segments
- Cells of dimension 2 are polygons

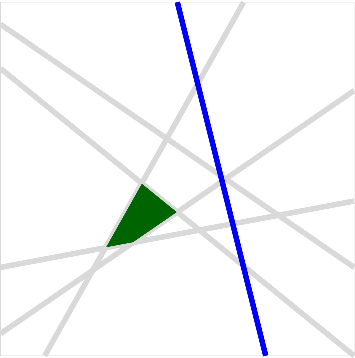


Example—Arrangements of lines

Definition

Given a finite collection S of geometric objects such as hyperplanes or spheres in $\mathbb{R}^d$, the arrangement $\mathcal{A}(S)$ is the decomposition of $\mathbb{R}^d$ into connected open cells of dimensions $0, 1, \ldots, d$ induced by S.

- S is a set of n lines in $\mathbb{R}^2$
- Cells of dimension 0 are points
- Cells of dimension 1 are segments
- Cells of dimension 2 are polygons

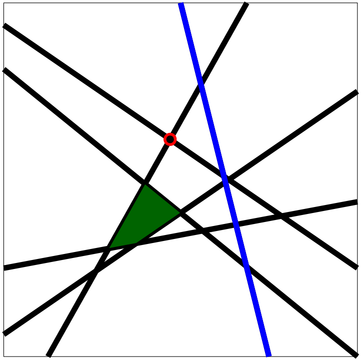


Example—Arrangements of lines

Definition

Given a finite collection S of geometric objects such as hyperplanes or spheres in $\mathbb{R}^d$, the arrangement $\mathcal{A}(S)$ is the decomposition of $\mathbb{R}^d$ into connected open cells of dimensions $0, 1, \ldots, d$ induced by S.

- S is a set of n lines in $\mathbb{R}^2$
- Cells of dimension 0 are points
- Cells of dimension 1 are segments
- Cells of dimension 2 are polygons



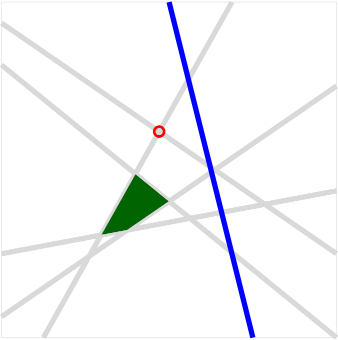
Complexity of an arrangement

Definition

The complexity of an arrangement $\mathcal{A}(S)$ is the overall number of cells (of all dimensions) of $\mathcal{A}(S)$.

E.g., for a planar arrangement, we count the total number of vertices edges and faces and get that the complexity is $O(n^2)$.

- # of vertices $n \cdot (n-1)$
- # of edges n^2
- Using Euler's formula $|V| - |E| + |F| = 2$ # of faces is $3n/2 + 1$



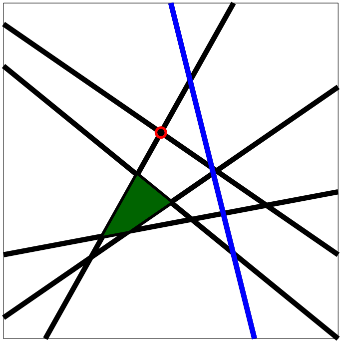
Complexity of an arrangement

Definition

The complexity of an arrangement $\mathcal{A}(S)$ is the overall number of cells (of all dimensions) of $\mathcal{A}(S)$.

E.g., for a planar arrangement, we count the total number of vertices edges and faces and get that the complexity is $O(n^2)$.

- $\#$ of vertices $n \cdot (n-1)$
- $\#$ of edges n^2
- Using Euler's formula
  $|V| - |E| + |F| = 2$
  $\#$ of faces is $3n/2 + 1$



Complexity of an arrangement (cont.)

We assume the general-position assumption: two lines meet in a single point, no three points are co-linear, etc.

Thm. 28.1.3

The maximal combinatorial complexity of an arrangement of n well behaved curves in the plane is $O(n^2)$; there are such arrangements whose complexity is $\Omega(n^2)$.

Thm. 28.1.4

The maximal combinatorial complexity of an arrangement of n well behaved curves in $\mathbb{R}^d$ is $O(n^d)$; there are such arrangements whose complexity is $\Omega(n^d)$.

Complexity of an arrangement (cont.)

We assume the general-position assumption: two lines meet in a single point, no three points are co-linear, etc.

Thm. 28.1.3

The maximal combinatorial complexity of an arrangement of n well behaved curves in the plane is $O(n^2)$; there are such arrangements whose complexity is $\Omega(n^2)$.

Thm. 28.1.4

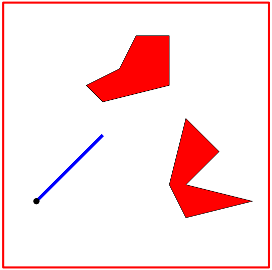
The maximal combinatorial complexity of an arrangement of n well behaved curves in $\mathbb{R}^d$ is $O(n^d)$; there are such arrangements whose complexity is $\Omega(n^d)$.

Arrangements & C-spaces

- Arrangements are used to exactly *discretize* a continuous problem without loss of information
- The general idea is that all objects within a k-dimensional cell have the same property
- This property only changes when we move from cell to cell
- The geometric objects, defining the arrangement are called critical curves

Consider a rod translating in the room

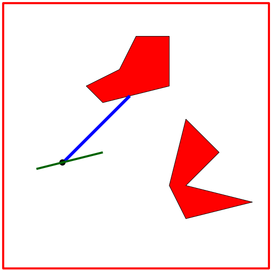
- The property considered is if the rod intersects obstacles
- It can only change it's property by touching an obstacle (semi-free places)
- This can happen when
    - a rod vertex touches an obstacle edge (v-e event)
    - a rod edge touches an obstacle vertex (e-v event)



Arrangements & C-spaces (cont.)

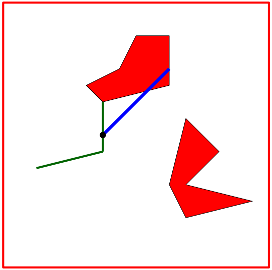
Consider a rod translating in the room

- The property considered is if the rod intersects obstacles
- It can only change it's property by touching an obstacle (semi-free places)
- This can happen when
  - a rod vertex touches an obstacle edge (v-e event)
  - a rod edge touches an obstacle vertex (e-v event)



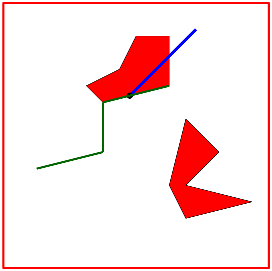
Consider a rod translating in the room

- The property considered is if the rod intersects obstacles
- It can only change it's property by touching an obstacle (semi-free places)
- This can happen when
    - a rod vertex touches an obstacle edge (v-e event)
    - a rod edge touches an obstacle vertex (e-v event)



Consider a rod translating in the room

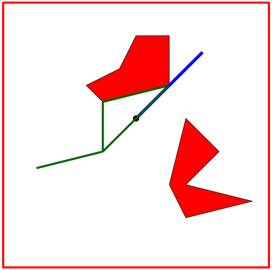
- The property considered is if the rod intersects obstacles
- It can only change it’s property by touching an obstacle (semi-free places)
- This can happen when
  - a rod vertex touches an obstacle edge (v-e event)
  - a rod edge touches an obstacle vertex (e-v event)



Arrangements & C-spaces (cont.)

Consider a rod translating in the room

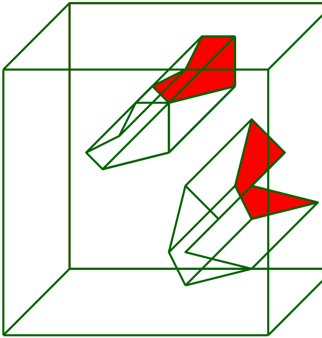
- The property considered is if the rod intersects obstacles
- It can only change it's property by touching an obstacle (semi-free places)
- This can happen when
  - a rod vertex touches an obstacle edge (v-e event)
  - a rod edge touches an obstacle vertex (e-v event)



Arrangements & C-spaces (cont.)

Consider a rod translating in the room

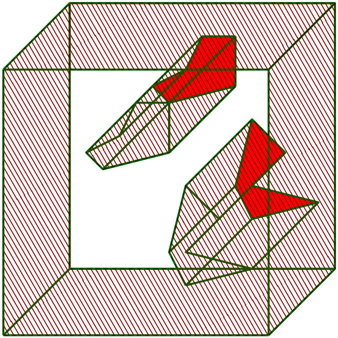
- The property considered is if the rod intersects obstacles
- It can only change it's property by touching an obstacle (semi-free places)
- This can happen when
  - a rod vertex touches an obstacle edge (v-e event)
  - a rod edge touches an obstacle vertex (e-v event)



Arrangements & C-spaces (cont.)

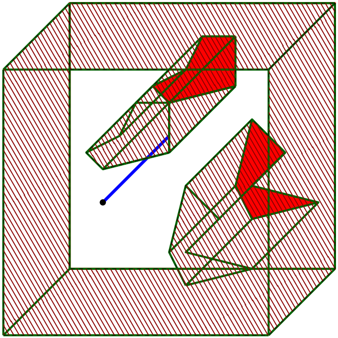
Consider a rod translating in the room

- The property considered is if the rod intersects obstacles
- It can only change it's property by touching an obstacle (semi-free places)
- This can happen when
  - a rod vertex touches an obstacle edge (v-e event)
  - a rod edge touches an obstacle vertex (e-v event)



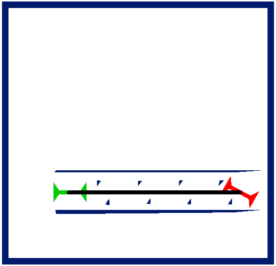
Consider a rod translating in the room

- The property considered is if the rod intersects obstacles
- It can only change it's property by touching an obstacle (semi-free places)
- This can happen when
  - a rod vertex touches an obstacle edge (v-e event)
  - a rod edge touches an obstacle vertex (e-v event)



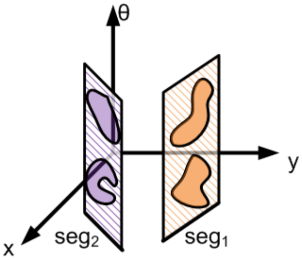
Application—Rotating Robot Along a Translation Segment

- Parameterization
  - $\alpha \in [0, 1]$ reference point on segment
  - $\theta \in [0, 2\pi]$ rotation angle
- Critical Curves are in the form of rational functions



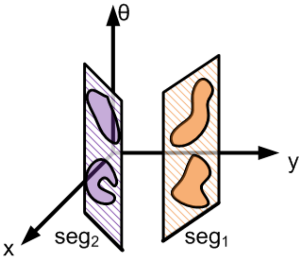
Application—Rotating Robot Along a Translation Segment

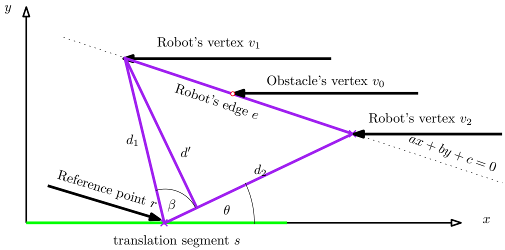
- Parameterization
  - $\alpha \in [0, 1]$ reference point on segment
  - $\theta \in [0, 2\pi]$ rotation angle
- Critical Curves are in the form of rational functions



Application—Rotating Robot Along a Translation Segment

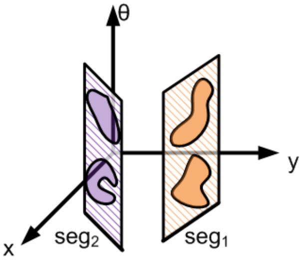
- Parameterization
  - $\alpha \in [0, 1]$ reference point on segment
  - $\theta \in [0, 2\pi]$ rotation angle
- Critical Curves are in the form of rational functions

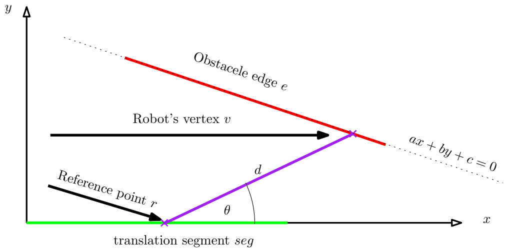




Application—Rotating Robot Along a Translation Segment

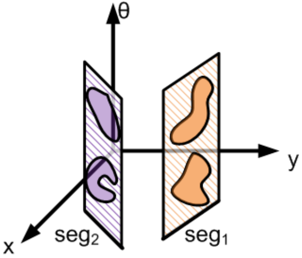
- Parameterization
  - $\alpha \in [0, 1]$ reference point on segment
  - $\theta \in [0, 2\pi]$ rotation angle
- Critical Curves are in the form of rational functions

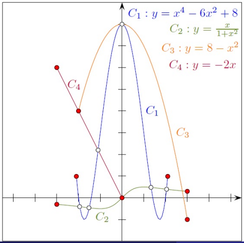


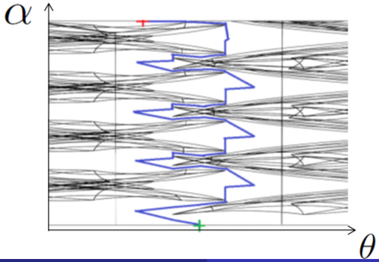


Application—Rotating Robot Along a Translation Segment

- Parameterization
  - $\alpha \in [0, 1]$ reference point on segment
  - $\theta \in [0, 2\pi]$ rotation angle
- Critical Curves are in the form of rational functions







General exact algorithms for motion planning

- The first general exact algorithm was proposed by Schwartz and Sharir [SS83]
- It assumes that the contact surfaces of $\mathcal{X}_{\text{free}}$ are explicitly defined by a total of n polynomials of maximum degree k.
- It uses the notion of Collins's cylindrical algebraic decomposition [C75] which is a recursive decomposition of $\mathcal{X}_{\text{free}}$ into cylindrical-like cells.
- These cells are used to construct a roadmap
- It is then easy to search for a collision-free path through this roadmap, if one exists, between the (cell containing the) initial robot placement and the (cell containing the) final placement.

General exact algorithms for motion planning—Cont.

Thm. 50.1.1

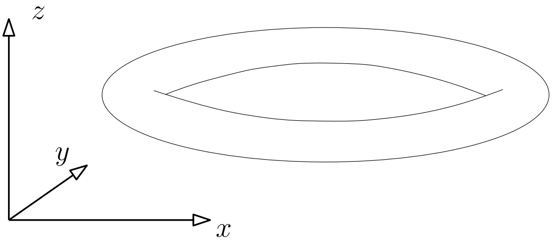
Any motion planning problem, with d degrees of freedom, for which the contact surfaces are defined by a total of n polynomials of maximum degree k, can be solved by Collins's cylindrical algebraic decomposition, in randomized expected time $O\left((nk)^{3^d}\right)$).

General exact algorithms for motion planning—Cont.

- The method by Schwartz and Sharir [SS83] that uses Collin's decomposition runs in time that is doubly exponential
- This was improved by Canny [C87] by multiple projections of $\mathbb{R}^d$ to $\mathbb{R}^2$ many times

Thm. 50.1.2

Any motion planning problem, as in 50.1.1 can be solved in $n^k \cdot (\log n) \cdot d^{O(k^4)}$ deterministic time.

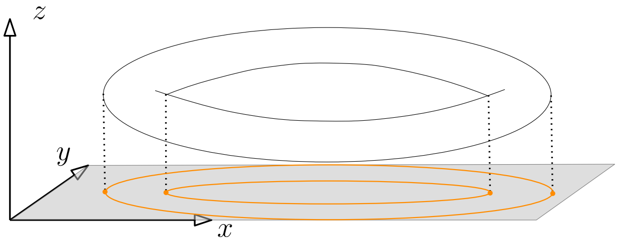


General exact algorithms for motion planning—Cont.

- The method by Schwartz and Sharir [SS83] that uses Collin's decomposition runs in time that is doubly exponential
- This was improved by Canny [C87] by multiple projections of $\mathbb{R}^d$ to $\mathbb{R}^2$ many times

Thm. 50.1.2

Any motion planning problem, as in 50.1.1 can be solved in $n^k \cdot (\log n) \cdot d^{O(k^4)}$ deterministic time.

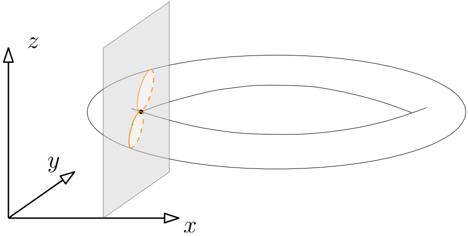


General exact algorithms for motion planning—Cont.

- The method by Schwartz and Sharir [SS83] that uses Collin's decomposition runs in time that is doubly exponential
- This was improved by Canny [C87] by multiple projections of $\mathbb{R}^d$ to $\mathbb{R}^2$ many times

Thm. 50.1.2

Any motion planning problem, as in 50.1.1 can be solved in $n^k \cdot (\log n) \cdot d^{O(k^4)}$ deterministic time.

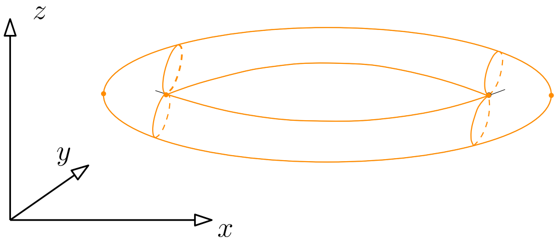


General exact algorithms for motion planning—Cont.

- The method by Schwartz and Sharir [SS83] that uses Collin's decomposition runs in time that is doubly exponential
- This was improved by Canny [C87] by multiple projections of $\mathbb{R}^d$ to $\mathbb{R}^2$ many times

Thm. 50.1.2

Any motion planning problem, as in 50.1.1 can be solved in $n^k \cdot (\log n) \cdot d^{O(k^4)}$ deterministic time.



General exact algorithms for motion planning—Cont.

- The method by Schwartz and Sharir [SS83] that uses Collin's decomposition runs in time that is doubly exponential
- This was improved by Canny [C87] by multiple projections of $\mathbb{R}^d$ to $\mathbb{R}^2$ many times

Thm. 50.1.2

Any motion planning problem, as in 50.1.1 can be solved in $n^k \cdot (\log n) \cdot d^{O(k^4)}$ deterministic time.

This was Canny's PhD which which won the ACM best thesis award.

His MSc work on edge detection is much more recognized ($> 33,000$ citations).

Further reading

- CG book, Chapter 50
- LaValle's book, Chapter 6.4